Protocole Ad hoc Proactif Anonyme à Base de Cryptographie Homomorphique

Antoine Guellier (antoine.guellier@insa-rennes.fr)*
Christophe Bidan (christophe.bidan@supelec.fr)*
Nicolas Prigent (nicolas.prigent@supelec.fr) *

Résumé : Avec l'avènement des systèmes ubiquitaires, la question du respect de la vie privée devient primordiale. Dû aux spécificités de ces systèmes, il est particulièrement difficile d'empêcher la fuite d'informations telles que l'identité des personnes, leur emplacement géographique, leurs relations, etc. Dans le monde des réseaux ad hoc, il est nécessaire que les protocoles de routage assurent la protection des identités des nœuds ainsi que leur position tant géographique que relative. Dans cet article, nous proposons un nouveau protocole de routage ad hoc *proactif* se basant sur la cryptographie homomorphique complète pour assurer la protection des informations privées des nœuds. Cette technologie offre de nouvelles possibilités : il n'est plus nécessaire de déchiffrer une information pour l'utiliser. Avec un minimum d'hypothèses, nous montrons qu'un protocole de routage ad hoc proactif garantissant un fort respect de la vie privée est possible.

Mots Clés : Ad hoc, Routage, Proactif, Anonyme, Homomorphique

1 Introduction

Avec l'avènement des systèmes ubiquitaires, la question du respect de la vie privée devient primordiale. La protection des informations privées diffère de la protection classique des données. Il s'agit ici d'empêcher la fuite d'informations comme l'identité des personnes, leur emplacement géographique, leurs relations, etc. Dans le monde des MANETs (*Mobile Ad hoc NETworks*, Réseaux Ad hoc Mobiles), cela implique que les protocoles de routage ad hoc doivent assurer la protection des identités des nœuds ainsi que leur position tant géographique que relative. Ces prérogatives sont nécessaires dans un contexte militaire, où un changement brusque dans l'organisation des communications peut révéler une action ou attaque imminente, mais ils sont aussi souhaitables dans le domaine civil, par exemple lorsque des utilisateurs de téléphones mobiles se connectent entre eux (de manière ad hoc) dans le but de partager des informations ou de jouer en ligne avec d'autres utilisateurs tout en restant anonymes. Il existe plusieurs solutions de routage ad hoc « anonyme » (i.e. respectueux de la vie privée), et la plupart proposent une approche *réactive*.

Les protocoles de routage réactifs utilisent des *requêtes de route* : lorsqu'un nœud désire communiquer avec une destination, il diffuse dans tout le réseau une requête à laquelle la destination répondra par une *réponse de route*. Ainsi dans ce type de routage, les nœuds gardent très peu d'informations sur la topologie. L'approche *proactive*, elle, maintient des

∗. Supélec, Avenue de la Boulaie, 35510 Cesson-Sévigné, FRANCE

tables de routage exhaustives concernant toutes les destinations : les nœuds connaissent donc la topologie complète du réseau.

La principale contribution de cet article est de proposer un tout nouveau protocole de routage proactif anonyme, APART, en posant le moins d'hypothèses possibles. *A priori* il semble qu'avec l'approche proactive il soit plus difficile d'assurer le respect de la vie privée des nœuds car chaque nœud dispose de beaucoup d'informations sur le réseau. Mais cette hypothèse peut être nuancée. En particulier l'utilisation de la cryptographie homomorphique complète [Gen10] offre de nouvelles possibilités : il n'est plus nécessaire de déchiffrer une information pour l'utiliser. Même si l'inefficacité pratique de cette technologie pose problème, et en particulier pour une application dans les réseaux ad hoc, nous nous contentons ici de montrer qu'un protocole anonyme proactif est possible grâce à elle.

Le reste du papier est organisé comme suit. La section 2 présente les travaux apparentés. La section 3 expose les hypothèses et modèles dans lesquels nous nous plaçons pour notre protocole, décrit en section 4. La section 5 conclut.

2 Travaux Similaires

Les travaux sur l'anonymat dans les réseaux traditionnels commencent avec les MIX-Nets de Chaum [CARC81]. Cette approche utilise des routeurs spécifiques appelés *MIX* qui chiffrent et déchiffrent les paquets afin de changer leur apparence et les réordonnent aléatoirement pour rendre le chemin emprunté intraçable. En particulier, grâce à ce mécanisme, la source du message est gardée secrète.

Une adaptation de cette approche, le routage en oignon [GRS99], a été reprise et adaptée par Kong *et al.* dans ANODR [KH03] pour concevoir un protocole ad hoc anonyme réactif. Les requêtes de route contiennent les informations de routage qui s'accumulent dans une structure organisée en couches (d'où le nom d'oignon) au fur et à mesure que la requête progresse : chaque nœud doit chiffrer une estampille avec une clé secrète et rediffuser la requête lorsqu'il la reçoit. Il reconnaîtra et déchiffrera cette estampille quand la réponse de route passera au retour. Malheureusement, les messages de ce protocole ont la fâcheuse propriété de grandir en taille le long de la route. De ce fait, il est facile de suivre un message, en particulier lorsqu'il y a peu de communications.

Il existe plusieurs autres protocoles de routage ad hoc respectueux de la vie privée, plus ou moins efficaces selon le but recherché et selon les compromis entre efficacité, anonymat et consommation d'énergie. ODAR [SCB06] est un protocole réactif qui cache efficacement l'identité des nœuds source, destination et relais en utilisant des filtres de Bloom [BGK+08]. Quand une requête de route est diffusée, chaque nœud qui la reçoit s'ajoute dans le filtre de Bloom associé à cette requête. Ainsi quand la réponse de route revient à la source, cette dernière dispose d'un filtre dans lequel les nœuds relais sont répertoriés. La source attachera ce filtre aux messages qu'elle enverra à la destination et les nœuds la long de la route pourront savoir qu'ils sont censés les relayer.

Le protocole MASK [ZLLF06] est un autre un protocole réactif, mais qui a la particularité d'utiliser un mécanisme proactif de découverte et d'authentification du voisinage. Lors du processus d'authentification, un nœud N ne révèle jamais sa vraie identité et utilise à la place des *pseudonymes* PS_N. À l'issue de l'authentification, chaque paire de nœuds voisins partage un ensemble *d'identifiants de liens* et de clés symétriques permettant de sécuriser les communications à 1 saut. Un identifiant de lien est un simple nombre aléa-

toire que deux nœuds utilisent pour s'adresser l'un l'autre, par exemple en l'insérant dans l'entête des paquets à la place de leurs adresses. Plusieurs identifiants de liens sont générés car un identifiant de lien n'est utilisé qu'une fois puis jeté pour éviter qu'un observateur extérieur ne puisse suivre l'évolution de ce lien dans l'espace et le temps. Nous réutilisons ce mécanisme dans notre proposition.

La principale différence de notre approche est que nous proposons une solution *proactive* utilisant la cryptographie homomorphique. Il existe en effet très peu de protocoles anonymes proactifs. Nezhad et al. ont travaillé dans ce sens et proposé *V-Routing* [NMMB09], un protocole proactif garantissant l'anonymat des routes et des nœuds en construisant un chemin triangulaire virtuel. Pour cela, ils s'appuient sur une découverte du réseau avec un protocole de routage ad hoc « standard ». Cependant, le modèle de réseau choisi suppose la présence de nœuds acceptant de révéler leur identité pour aider les autres à cacher la leur. Les mêmes auteurs proposent une technique pour découvrir de manière anonyme la topologie d'un réseau de capteurs [NMM07]. Ce type de réseau comprend un nœud particulier appelé « point de collecte » qui centralise les informations sur la topologie du réseau. Ce nœud connaît potentiellement l'identité de tous les nœuds et est en contradiction avec les propriétés d'un réseau ad hoc.

3 Modèles et Hypothèses

3.1 Modèle de réseau

Nous nous plaçons dans le contexte des MANETs, où les nœuds communiquent par WiFi avec une portée limitée. Nous ne considérons que les liens bidirectionnels (i.e. si X peut entendre Y, alors Y peut entendre X). Chaque nœud possède un identifiant unique [1]. Si un nœud a plusieurs interfaces réseau, il en choisit une comme interface principale. Nous nous plaçons au dessus des protocoles de couche 2 ou 3 et nous ne nous intéressons pas à la gestion de l'anonymat pour ces couches. Nous préciserons seulement que toute la pile de protocole doit protéger l'identité des nœuds. En particulier, nous supposons les nœuds capables de fonctionner en « mode promiscuité » pour écouter tous les paquets passant à leur portée, et que les adresses MAC sont fixées à l'adresse de diffusion.

Enfin, nous faisons l'hypothèse qu'avant le déploiement du réseau, les nœuds désirant communiquer se sont échangés leurs identifiants « réels » et clés publiques *permanentes* respectives. En effet, chaque nœud X dispose de plusieurs types de clés publiques : une unique clé permanente PK_X^{perm}, ainsi que plusieurs clés publiques PK_X (autant que nécessaire) qu'il utilise successivement. Ces dernières ne sont pas liées à son identité ID_X. Chaque nœud X peut aussi générer des clés à usage unique, notées PK_X^{tmp} et SK_X^{tmp}.

3.2 Modèle d'attaquant

Il est possible de faire la distinction entre plusieurs types d'attaquants, combinables entre eux :

- *Actif/Passif* : injecte ou modifie des paquets / se contente d'espionner les communications et de déduire des informations sur le réseau ;
- *Interne/Externe* : possède / ne possède pas les paramètres cryptographiques « secrets » du réseau (e.g. une clé secrète partagée par tous les nœuds du réseau) ;

1. Une solution pour implémenter ces identifiants est d'utiliser des identifiants SUCV [MC02]

ID_A	identité du nœud A	$E_{PK_A}(M)$	chiffré *homomorphique* de M avec PK_A
PK_A^{perm}	la clé permanente de A	L_{AB}	identifiant de lien entre A et B
PK_A	une des clés publiques de A	K_{AB}	clé secrète associée à L_{AB}
SK_A	une des clés privées de A	$NbSauts$	longueur d'une route (en nb de sauts)
PK_A^{tmp}	une clé publique temporaire de A	g_A	nombre secret de A, membre de $\mathbb{Z}_q^*$
$Alea_A$	nombre aléatoire généré par A	$LocalID_A^D$	identifiant de A pour la destination D
PS_A	un des pseudonymes de A	$\mathbb{G}_1$	groupe additif d'ordre premier q
TD_A	Table des Destination de A	$\mathbb{G}_2$	groupe multiplicatif d'ordre premier q
TR_A	Table des Relais de A	H_A	fonction de hachage de A : $\{0,1\}^* \rightarrow \mathbb{G}_1$
$K(M)$	chiffré de M avec la clé K	$\hat{e}$	application bilinéaire $\mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$

Table 1: Terminologie et Notations

– *Local/Global* : écoute les communications WiFi dans une zone délimitée / dans tout le réseau à la fois.

Dans cet article, nous considérons le modèle d'attaquant externe (actif ou passif, local ou global) et assurons un anonymat contre ce type d'attaquant. Les attaquants internes (i.e. un attaquant externe qui a réussi à *compromettre* un nœud) étant bien plus difficiles à gérer, nous emploierons le modèle « honnête mais curieux » : l'attaquant interne essaiera d'accumuler le plus d'informations possible sur le réseau, mais restera conforme au protocole.

3.3 Anonymat visé

Nous nous référons à Pfitzmann et Kohntopp [PK01] pour définir les concepts d'anonymat des entités et d'« inassociabilité » (*unlinkability*) des événements. Notre protocole se fixe les objectifs suivants :

– *Anonymat de la source, de la destination et de la route* : il n'est pas possible d'identifier la source ou la destination parmi un ensemble de nœuds, possiblement le réseau entier. L'anonymat de la route peut être réduit à l'anonymat des nœuds relais.
– *Inassociabilité source/message* : à partir d'un message donné, il n'est pas possible d'identifier sa source.
– *Inassociabilité destination/message* : à partir d'un message donné, il n'est pas possible d'identifier sa destination.
– *Inassociabilité source/destination* : savoir qui communique avec qui est impossible.
– *Non-localisation* : trouver la position géographique ou relative des nœuds est impossible.

Notons que dans notre proposition, nous n'essayons pas de garantir la confidentialité, l'intégrité ou l'authenticité des données échangées entre communicants. En effet, APART fournit seulement une couche de communication anonyme, et si nécessaire, ces propriétés peuvent être implémentées dans une couche supérieure (par ex. en utilisant SSL ou TLS).

4 Le protocole APART

Le protocole de routage APART divise le routage en 2 composantes, séparant l'activité de routage des nœuds en deux. La première composante constitue l'initialisation de route, la deuxième le relais « aveugle ». Ce relais est aveugle dans le sens où les nœuds relaient un message sans savoir qui est sa destination. De manière générale, les tables de routage des

nœuds ne donnent d'ailleurs aucune information explicite sur les destinations du réseau. C'est pourquoi l'initialisation de route exige un mode opératoire particulier (c.f. § 4.3).

Par la suite, nous utilisons le système d'authentification de voisinage de MASK comme décrit en section 2. Pour plus de simplicité, nous noterons L_{NM} l'identifiant de lien entre deux nœuds N et M. Même si N et M partagent bien *plusieurs* identifiants de liens, nous en faisons abstraction et considérons que chaque lien L^i_{NM} n'est utilisé qu'une fois puis remplacé par son successeur L^{i+1}_{NM} De même pour les clés symétriques, notées K_{NM}. Les pseudonymes d'un nœud N sont aussi désignés avec une unique notation, PS_N. Les notations utilisées sont données dans la table 1.

4.1 Tables de routage

Chaque nœud maintient deux tables de routage : une *Table des Destinations* (TD) et une *Table des Relais* (TR). La première répertorie toutes les destinations et routes connues. La deuxième est orientée relais et c'est elle qu'utilisent les nœuds relais pour relayer les messages. Une entrée de la TD du nœud P est de la forme

$$\langle\ LocalID_{Dest},\ Dest,\ LongueurRoute,\ CléPubliqueDest\ \rangle$$

En particulier, pour une destination D, $\langle\ LocalID^P_D,\ E_{PK_D}(ID_D),\ E_{PK_D}(NbSauts),\ PK_D\ \rangle$, où $LocalID^P_D$ est une valeur générée conjointement par P et D, servant à P pour identifier de manière unique la destination D. PK_D est *une* des clés publiques de D et, au contraire de sa clé permanente PK^{perm}_D, n'est pas liée à son identité. Une entrée de la TR du nœud P est de la forme

$$\langle\ Lien_{amont},\ LocalID_{Dest},\ Lien_{aval}\ \rangle$$

En particulier une entrée concernant une route vers D sur laquelle P se trouve est $\langle\ L_{XP},\ LocalID^P_D,\ L_{PY}\ \rangle$, avec X le nœud précédent, en amont de P sur la route vers D, et Y le suivant (en aval).

4.2 Découverte de la topologie

Proposition de route

Ces tables TD et TR sont remplies lors de la découverte de la topologie. Notre paradigme est diamétralement opposé aux protocoles réactifs qui utilisent des *requêtes* de routes. Ici la topologie est diffusée par *propositions* de routes : quand un nœud connaît une route, il propose à ses voisins directs de relayer leurs messages vers la destination de la route. Les routes connues par un nœud sont dans sa TD. En début de vie du réseau, cette table ne contient que le nœud lui même. Chaque nœud commence donc par proposer une route vers lui-même à ses voisins, qui vont eux même proposer cette route à leurs voisins, etc. Ainsi les routes vont s'agrandir au fur et à mesure jusqu'à atteindre les bords du réseau.

Lorsqu'un nœud N propose une route pointant vers la destination D, il diffuse localement le message suivant :

$$\langle\ PS_N,\ Alea_N,\ E_{PK_D}(ID_D),\ E_{PK_D}(NbSauts),\ PK_D\ \rangle \quad (1)$$

Avec PS_N un pseudonyme de N et $Alea_N$ un nombre aléatoire, deux paramètres utiles à l'authentification et à la génération d'identifiants de liens et de clés. L'identité ID_D de

D est chiffrée de manière probabiliste avec une clé publique de D par lui même. Cette valeur (ainsi que PK_D) est donnée en premier lieu par D à ses voisins lorsqu'il propose une route vers lui-même et elle va se répandre ensuite dans le réseau. Ainsi aucun nœud excepté D lui-même n'accède à la valeur ID_D. De même pour $NbSauts$, excepté que cette valeur sera incrémentée homomorphiquement par chaque nœud acceptant la route.

Acceptation/Refus de route

Les voisins de N recevant l'offre de route vont se servir de PK_D et $E_{PK_D}(NbSauts)$ pour décider s'ils acceptent ou non la route. Cette décision est en partie *inconsciente* car elle est prise automatiquement en appliquant une fonction de décision (déterministe) sur le chiffré. Grâce aux propriétés du chiffrement homomorphique complet, les traitements sur $E_{PK_D}(NbSauts)$ seront reportés sur $NbSauts$. Le processus de décision pour un nœud P recevant une offre se déroule ainsi :

1. Si c'est la première fois qu'il voit la clé PK_D le receveur P accepte la route car D est potentiellement une nouvelle destination qu'il ne connaît pas.
2. S'il a déjà assez de routes vers D (selon un certain seuil à définir), il refuse la route. Pour compter le nombre de route qu'il possède vers D, un nœud utilise sa TD.
3. En dernier lieu, le receveur P applique un traitement homomorphique sur la longueur de la route de manière à évaluer $(E_{PK_D}(NbSauts) \leq E_{PK_D}(NbSauts_{MAX}))$. Pour cela, il suffit de calculer $E_{PK_D}(NbSauts) - E_{PK_D}(NbSauts_{MAX})$ et de prendre le chiffré du bit de poids fort. Le résultat sera $E_{PK_D}(1)$ si $NbSauts - NbSauts_{MAX} < 0$.

Cette fonction rend $E_{PK_D}(b)$ avec $b = 1$ si la route est acceptée et $b = 0$ sinon. Bien sûr un nœud refuse la route si celle-ci pointe vers lui-même. Il peut s'en rendre compte en vérifiant si PK_D lui appartient ou non. D'autre part, un nœud n'accepte qu'une fois une même proposition, identifiable par le couple (PS_N, PK_D) par exemple.

Réponse à l'offre de route

Remarquons qu'à ce moment, P ne sait toujours pas s'il a accepté la route ou non. Il faut déchiffrer $E_{PK_D}(b)$ pour obtenir le résultat. Or seul D est en mesure de le faire. Il va donc falloir que P envoie ce résultat à N qui le transmettra à D par la route dont il dispose vers D. Mais avant cela, P applique un autre traitement au chiffré de ID_D : une fonction arbitraire *à sens unique* déterministe H_P à valeur dans $\mathbb{G}_1$ (par ex. une fonction de hachage). La valeur retournée en résultat appartient à $\mathbb{G}_1$ et est notée $PreLocalID_D^P$. P renvoie alors ces deux chiffrés à N dans le message suivant (avec PS_P et $Alea_P$ utiles à l'authentification de P et N, et PK_P^{tmp} une clé à usage unique générée par P qui servira à chiffrer la réponse) :

$$\langle\ PS_P,\ Alea_P,\ E_{PK_D}(b),\ E_{PK_D}(PreLocalID_D^P),\ PK_P^{tmp}\ \rangle \tag{2}$$

Le nœud N va ensuite transmettre $E_{PK_D}(b)$ et $E_{PK_D}(PreLocalID_D^P)$ à D à travers les nœuds $X_1, X_2, \ldots, X_n$ qui vont les relayer de proche en proche en utilisant les identifiants de liens et clés symétriques générés lors de la création de la route. Le message allant de N à D sera de la forme :

$$\langle\ L_{X_kX_{k+1}},\ K_{X_kX_{k+1}}(Alea'_N,\ PK_N^{tmp},\ E_{PK_D}(b),\ E_{PK_D}(PreLocalID_D^P), PK_P^{tmp})\ \rangle \tag{3}$$

Avec $k \in [0, n]$ et $X_0 = N$ et $X_{n+1} = D$. $Alea'_N$ est un nouveau nombre aléatoire généré par N qui lui sera utile pour reconnaître la réponse de D, et PK_N^{tmp} une clé à usage

unique qui servira à chiffrer une partie de la réponse. On remarque que grâce au chiffrement/déchiffrement avec les clés $K_{X_k X_{k+1}}$, le message n'est pas traçable par un observateur extérieur lors de son acheminement dans le réseau.

Quand D obtient les deux chiffrés $E_{PK_D}(b)$ et $E_{PK_D}(PreLocalID_D^P)$, il les déchiffre et les renvoie au proposeur N. Auparavant, il applique un post-traitement à la valeur $PreLocalID_D^P$ en la multipliant par un nombre secret $g_D \in \mathbb{Z}_q^*$ (q premier) qu'il aura choisit à la mise en place du réseau. Il obtient $LocalID_D^P = g_D \times PreLocalID_D^P$. Cette dernière donnée sera utilisée lors de l'initialisation de route. Les valeurs $LocalID_D^P$ et b sont respectivement chiffrées avec PK_P^{tmp} et PK_N^{tmp} pour que seuls P et N prennent connaissance de leur valeur. Elles sont ensuite renvoyées le long de la route *inversée* jusqu'à N (la route étant *en temps normal* unidirectionnelle de N vers D, les nœuds relais auront pris soin de noter leur prédécesseur sur cette route pour l'occasion). Le message de retour est de la forme :

$$\langle\, L_{X_{k+1}X_k},\ K_{X_{k+1}X_k}(Alea'_N,\ PK_N^{tmp}(b),\ PK_P^{tmp}(LocalID_D^P))\,\rangle \tag{4}$$

N reconnaîtra le paquet grâce au nombre aléatoire $Alea'_N$. S'il reçoit $b = 0$, il sait que la proposition a été refusée. Si $b = 1$, P a accepté la route. N l'en informe et lui communique $LocalID_D^P$ (chiffré avec PK_P^{tmp}). P crée alors une nouvelle entrée dans sa TD avec cette valeur de $LocalID_{Dest}$ ainsi que les chiffrés de ID_D et $NbSauts$ et la clé PK_D dont il dispose déjà. N crée une nouvelle entrée dans sa TR avec $Lien_{amont} = L_{PN}$, $Lien_{aval} = L_{NX_1}$ et $LocalID_{Dest} = LocalID_D^N$ (il trouve cette dernière valeur dans l'entrée de sa TD qu'il est en train de proposer). La figure 1 résume les différents messages échangés lors d'une proposition de route.

Champ LocalID$_{\text{Dest}}$

Avec la valeur $LocalID_D^P$, P dispose d'un identifiant qu'il ne peut lier à aucune identité mais qui lui permet de compter le nombre de routes connues vers la destination D (en comptant le nombre d'entrées de sa TD ayant la valeur $LocalID_D^P$). De plus, la manière dont la valeur $LocalID_D^P$ est construite va permettre l'initialisation de route. En effet, en l'état, les tables TD et TR ne donnent aucune indication explicite sur la destination de la route et une communication ne peut être initialisée trivialement.

De son côté, D a aussi connaissance de la valeur de $LocalID_D^P$ mais ne sait pas qui en a l'usage. Au mieux, quand D reçoit un $PreLocalID_D^P$ à traiter, il sait que *quelqu'un* a accepté (ou refusé) une route vers lui, et il peut compter le nombre de route que celui-ci a accepté car il reconnaît la valeur de $PreLocalID_D^P$ quand il la reçoit. En effet, pour un nœud P donné, $PreLocalID_D^P$ et $LocalID_D^P$ ne changent pas au cours du temps. Cependant cela ne porte en aucun cas atteinte à la vie privée de P ou de quiconque.

De plus, comme le problème du logarithme discret est supposé difficile dans $\mathbb{G}_1$, il n'est

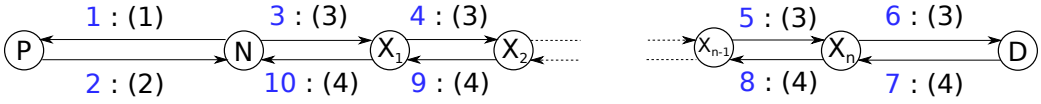

Figure 1: Paquets échangés durant une proposition de route (les premiers chiffres donnent l'ordre des messages, ceux entre parenthèse font référence aux messages (1), (2), (3) et (4))

pas possible de retrouver g_D à partir de $LocalID_D^P$ et $PreLocalID_D^P$. Enfin, l'identité ID_D de D ne peut être déduite de $PreLocalID_D^P$ du fait que H_P est une fonction à sens unique.

4.3 Routage

Une fois les tables TD et TR remplies, les nœuds peuvent commencer à communiquer entre eux. Le processus de routage se divise en deux étapes : l'initialisation de la route et le relais jusqu'à la destination. La première étape correspond à trouver un premier identifiant de lien valide vers la destination. Cependant, au vu des tables de routage qui ne laissent aucune indication sur l'identité des destinations, un mécanisme pour trouver ce premier « saut » est nécessaire.

Initialisation de route

Une source S désirant engager une communication avec une destination D va avoir besoin de l'aide d'un voisin V. En coopérant avec S, V peut trouver le $LocalID^V$ qui intéresse S, sans pour autant savoir quelle est la destination réelle D. S obtient, à travers V, une route valide vers D mais ne sait pas lequel de ses $LocalID^S$ y est associé. Si c'était le cas, quand S relaierait un message à destination de D pour le compte d'un autre nœud, il saurait tout de suite que ce message est destiné à D, ce qui est une entorse à la propriété d'inassociabilité destination/message. C'est pour cette raison que les Tables des Destinations ne doivent pas donner d'indication explicite sur l'identité des destinations.

Au préalable nous supposons que S connaît l'identité ID_D de D. Pour cela les nœuds qui souhaitent communiquer peuvent s'échanger leurs identités et clés publiques permanentes avant le déploiement du réseau. En vue de trouver un voisin pour l'assister, S diffuse localement une demande d'aide d'initialisation de route. Le premier voisin V qui répond à la demande sera celui qui assistera S. Nous supposons que S et V disposent tous deux d'une route vers D (sans savoir qu'elle pointe vers D) puisque en principe ils ont connaissance de tout le réseau.

En premier lieu, S et V s'authentifient avec un nouveau pseudonyme chacun et utilisent les identifiants de liens et clés partagées générées pour tous les échanges qui suivent. S commence par communiquer à V un ensemble de chiffrés homomorphiques chiffrés avec une clé à usage unique PK_S^{tmp} qu'elle aura généré pour l'occasion (et qu'elle donne également à V) : $E_{PK_S^{tmp}}(ID_D)$, $E_{PK_S^{tmp}}(PreLocalID_D^S)$ et pour *chaque entrée* X de sa TD, $E_{PK_S^{tmp}}(LocalID_X^S)$. De son côté, V va calculer $E_{PK_S^{tmp}}(PreLocalID_D^V)$ à partir de $E_{PK_S^{tmp}}(ID_D)$ et chiffrer pour *chaque entrée* Y de sa TD $E_{PK_S^{tmp}}(LocalID_Y^V)$.

Rappelons maintenant que $PreLocalID \in \mathbb{G}_1$ et donc $LocalID \in \mathbb{G}_1$ puisque $g \in \mathbb{Z}_q^*$. Nous nous donnons une application bilinéaire $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ telle que :

$$\hat{e}(LocalID_D^S, PreLocalID_D^V) = \hat{e}(PreLocalID_D^S, LocalID_D^V)$$

Nous supposons le lecteur familier avec le concept de *pairing* en cryptographie [Pat02]. Ainsi, V dispose de suffisamment d'éléments pour effectuer une recherche exhaustive sur $LocalID_X^S$ et $LocalID_Y^V$ jusqu'à trouver (X, Y) tel que le test

$$\begin{gathered}\hat{e}(E_{PK_S^{tmp}}(LocalID_X^S), E_{PK_S^{tmp}}(PreLocalID_D^V)) \\ = \\ \hat{e}(E_{PK_S^{tmp}}(PreLocalID_D^S), E_{PK_S^{tmp}}(LocalID_Y^V))\end{gathered}$$

rende *vrai*. Plus exactement, V doit itérer sur X et Y jusqu'à ce que $X = D$ et $Y = D$. Il est contraint de tester toutes les possibilités car ni S ni V ne savent (et ne doivent savoir) quel est le $LocalID_{Dest}$ de leur table respective qui correspond à D. Pour être plus précis, c'est V qui effectue les tests mais il doit donner le résultat chiffré à S pour qu'il le déchiffre avec sa clé privée.

Par ce procédé, V peut donc trouver $LocalID_D^V$ et relayer les messages de S vers D sans même savoir que D est la destination. Précisons qu'une source doit changer régulièrement de voisin d'initialisation de route V, y compris lors d'une même communication. Cela évite que V reste trop longtemps le relais par lequel passent tous les messages de S vers D. Cette position privilégiée ne doit pas être autorisée sur le long terme car elle permet de lier deux messages d'une même communication (sans forcément savoir qui sont les communicants), ce qui est une atteinte à la propriété d'inassociabilité message/message.

Relais

Une fois la route initialisée, S utilise l'identifiant de lien partagé avec V comme son premier « saut » vers D. De son côté, V utilise un des liens dont il dispose vers le $LocalID_D^V$ trouvé lors de l'initialisation de route. Les nœuds suivants sur la route que V utilise relaient simplement le paquet grâce à leur TR. Un message de données est du format $\langle\ L_{aval},\ \ K_{aval}(MESSAGE)\ \rangle$. Quand un nœud reçoit un paquet, il étudie la valeur L_{aval} et regarde si une des valeurs du champ $Lien_{amont}$ de sa TR y correspond. Si ce n'est pas le cas, il jette le paquet. Sinon, c'est qu'il doit relayer le paquet. Pour cela il note le $LocalID_{Dest}$ associé au $Lien_{amont}$ trouvé et il choisit au hasard dans sa TR un identifiant de lien $Lien_{aval}$ parmi ceux associés à cette valeur de $LocalID_{Dest}$. Il remplace le champ L_{aval} du paquet par son $Lien_{aval}$ et rediffuse localement le paquet. En outre, chaque paquet est chiffré avec la clé partagée associée à l'identifiant de lien L_{aval}. Un nœud relais doit déchiffrer le paquet avec cette clé K_{aval}, puis le re-chiffrer avec la clé associée au nouvel identifiant de lien. Ainsi, de proche en proche, le message est acheminé de manière sécurisée et intraçable pour un attaquant extérieur. De plus la route empruntée par un message n'est pas déterminée à l'avance et les chemins ont la bonne propriété d'être *partiellement disjoints*.

Quand la destination recevra le message, elle saura qu'il lui est destiné car l'identifiant de lien avec lequel elle le recevra correspondra à un identifiant de lien pour une route vers elle-même (i.e. une « fin » de route).

4.4 Autres composants

Le protocole APART intègre plusieurs autres caractéristiques et améliorations. Nous les énumérons ici brièvement.

Gestion des liens et de la mobilité Du fait que les nœuds bougent dans le réseau, la topologie change constamment. Autrement dit, les liens se forment et se cassent en permanence. Étant donné que les nœuds possèdent en général plusieurs routes vers une même destination, ils disposent la plupart du temps d'au moins un $Lien_{aval}$ vers n'importe quelle destination. De plus, la présence du champ $LocalID_{Dest}$ dans la TR d'un nœud va lui permettre de se comporter comme un MIX de Chaum [CARC81] et de réparer les routes : si un lien aval casse, le nœud peut en utiliser un autre associé à la même valeur de $LocalID_{Dest}$. S'il arrive qu'un nœud perde toutes ses routes vers un même $LocalID_{Dest}$ il en informe tous les $Lien_{amont}$ associés à ce $LocalID_{Dest}$ avec un message LINK_ERR.

Gestion des boucles La gestion des boucles est importante : il peut en apparaître beaucoup et il faut les prévenir. Pour cela nous utilisons des filtres de Bloom [BGK+08]. Chaque nœud maintient un filtre par destination (i.e. par $LocalID_{Dest}$) dont il dispose. Ce filtre contient l'identité hachée de chaque nœud qui se trouve sur la route. Plus précisément, comme une route se scinde potentiellement en plusieurs routes au niveau de chaque nœud, le filtre maintenu par un nœud N contient *tous* les nœuds de *toutes* les routes au départ de N vers la destination. Ainsi quand un nœud propose une route, il inclut ledit filtre dans le message de proposition (1). Si un nœud se voit dans le filtre, il refuse immédiatement la route car son acceptation créerait une boucle. Si un nœud accepte une route, il se rajoute dans le filtre de la proposition et fait une union avec le filtre qu'il possédait déjà pour le même $LocalID_{Dest}$ (s'il en possède un).

Gestion des pseudonymes et Anonymat Une composante qui apparaît en filigrane dans le protocole est la gestion des pseudonymes. Celle-ci est cruciale : il faut réaliser un compromis entre *l'anonymat* (changer souvent de pseudonyme) et *la perte de routes et de liens* (un changement de pseudonyme entraîne un changement d'identifiant de lien, donc un lien qui casse). En bref, un nœud utilise un pseudonyme différent pour chaque route qu'il propose, et il change régulièrement de pseudonyme d'offre de route vers lui même (et se re-propose immédiatement, sinon *toutes* les routes vers ce nœud sont supprimées et il disparaît momentanément du réseau). Il change aussi régulièrement de clé publique, sinon celle-ci l'identifie de manière unique dans tout le réseau. Enfin un nœud doit aussi de temps en temps changer de nombre secret g afin qu'il change de $LocalID$ auprès des autres nœuds : une donnée ne doit jamais identifier un nœud de manière permanente.

5 Conclusion

Les problèmes relatifs au respect de la vie privée dans les réseaux ad hoc sont encore difficiles à appréhender, d'autant que des compromis entre l'efficacité, l'anonymat et la sécurité sont nécessaires.

Dans cet article nous avons pris le parti de privilégier l'anonymat et nous avons réussi à prouver qu'un protocole proactif respectueux de la vie privée est possible en utilisant la cryptographie homomorphique. Cette technologie est le point central de notre protocole et c'est elle qui permet un anonymat fort. Bien sûr, en pratique les performances du protocole présenté ne sont pas adaptées aux exigences en terme de performances des réseaux ad hoc, et notre solution n'est une preuve de concept. Notons tout de même que depuis sa création en 2009, les performances de la cryptographie homomorphique complète ont bien évolué et cette technologie reste un centre d'intérêt majeur dans le monde de la cryptographie. De plus, il serait intéressant d'envisager l'utilisation de la cryptographie homomorphique *partielle*, plus simple et bien plus efficace (par ex. le schéma de Pailler [Pai99]).

Une simulation du protocole est en cours dans les locaux de Supélec, une analyse formelle du protocole est en projet, et les travaux futurs se concentreront sur une amélioration de l'anonymat et de la sécurité, voire des performances.

Références

[BGK+08] Prosenjit Bose, Hua Guo, Evangelos Kranakis, Anil Maheshwari, Pat Morin, Jason Morrison, Michiel Smid, and Yihui Tang. On the false-positive rate of

bloom filters. *Inf. Process. Lett.*, 108(4) :210–213, oct 2008.

[CARC81] David Chaum, Communications Of The Acm, R. Rivest, and David L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24 :84–88, 1981.

[Gen10] Craig Gentry. Computing arbitrary functions of encrypted data. *Communications of the ACM*, 53(3) :97–105, March 2010.

[GRS99] David Goldschlag, Michael Reed, and Paul Syverson. Onion routing. *Commun. ACM*, 42(2) :39–41, February 1999.

[KH03] Jiejun Kong and Xiaoyan Hong. Anodr : anonymous on demand routing with untraceable routes for mobile ad-hoc networks. In *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc '03, pages 291–302, New York, NY, USA, 2003. ACM.

[MC02] Gabriel Montenegro and Claude Castelluccia. Statistically unique and cryptographically verifiable (sucv) identifiers and addresses. In *20th Annual Network and Distributed System Security Symposium*, 2002.

[NMM07] Alireza A. Nezhad, Dimitris Makrakis, and Ali Miri. Anonymous topology discovery for multihop wireless sensor networks. In *Proceedings of the 3rd ACM workshop on QoS and security for wireless and mobile networks*, Q2SWinet '07, pages 78–85, New York, NY, USA, 2007. ACM.

[NMMB09] AlirezaA. Nezhad, Ali Miri, Dimitris Makrakis, and LuisO. Barbosa. Privacy within pervasive communications. *Telecommunication Systems*, 40 :101–116, 2009.

[Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Proceedings of the 17th international conference on Theory and application of cryptographic techniques*, EUROCRYPT'99, pages 223–238, Berlin, Heidelberg, 1999. Springer-Verlag.

[Pat02] K.G. Paterson. Cryptography from pairings : a snapshot of current research. *Information Security Technical Report*, 7(3) :41–54, 2002.

[PK01] Andreas Pfitzmann and Marit Köhntopp. Anonymity, unobservability, and pseudonymity — a proposal for terminology. In Hannes Federrath, editor, *Designing Privacy Enhancing Technologies*, volume 2009 of *Lecture Notes in Computer Science*, pages 1–9. Springer Berlin Heidelberg, 2001.

[SCB06] D. Sy, R. Chen, and L. Bao. Odar : On-demand anonymous routing in ad hoc networks. In *Mobile Adhoc and Sensor Systems (MASS), 2006 IEEE International Conference on*, pages 267–276. IEEE, 2006.

[ZLLF06] Yanchao Zhang, Wei Liu, Wenjing Lou, and Yuguang Fang. Mask : anonymous on-demand routing in mobile ad hoc networks. *Wireless Communications, IEEE Transactions on*, 5(9) :2376–2385, 2006.